(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 427 465 A2**

(12) EUROPEAN PATENT APPLICATION

(21) Application number: 90312005.3

(22) Date of filing: 01.11.90

(51) Int. Cl.5: G07F 7/10, G07C 9/00, G06F 1/00

(30) Priority: 09.11.89 US 433821

(43) Date of publication of application: 15.05.91 Bulletin 91/20

(84) Designated Contracting States: DE FR GB IT

(71) Applicant: AMERICAN TELEPHONE AND TELEGRAPH COMPANY
550 Madison Avenue
New York, NY 10022(US)

(72) Inventor: Claus, David Michael
7660 Brookview Lane
Indianapolis, Indiana 46250(US)
Inventor: Coutinho, Roy S.
10905 Timber Lane
Carmel, Indiana 46032(US)
Inventor: Murphy, Kevin Dean
6021 Middle Drive
Indianapolis, Indiana 46236(US)
Inventor: Snavley, James Damon
262 North Brewer Street
Greenwood, Indiana 46142(US)
Inventor: Zempol, Kenneth Robert
44 Center Grove Road, Apt. F26
Randolph, New Jersey 07920(US)

(74) Representative: Watts, Christopher Malcolm Kelway et al
AT&T (UK) LTD. AT&T Intellectual Property Division 5 Mornington Road
Woodford Green Essex IG8 OTU(GB)

(54) Databaseless security system.

(57) An improved security system, including a portable smart card (500) and a host computer (600), eliminates the need for the computer to store individual personal identification (ID) numbers for each user seeking access to the computer. Instead, the computer stores a first encryption algorithm E_1 used in converting a particular identification number $(ID)_n$ into a secret code S_n for that particular user. S_n also exists within the memory of the smart card having been loaded into its memory at the time of issue. A challenge number C is generated by the computer and transmitted to the smart card. Within the smart card and the computer, microprocessors respond to the challenge number C, the secret code S_n, and a second encryption algorithm E_2 in order to generate response numbers R_n and R'_n respectively. Thereafter, R_n is transmitted to the computer where it is compared with R'_n. A favorable comparison is necessary for gaining access to the computer.

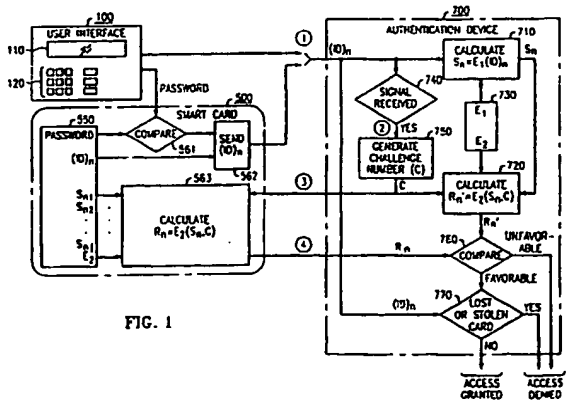


FIG. 1

EP 0 427 465 A2

DATABASELESS SECURITY SYSTEM

Technical Field

The present invention relates to a system for granting access to a secure facility, and more particularly to an authentication procedure.

Background of the Invention

A challenge for those who provide secure facilities is to exclude all unauthorized persons seeking entry while simultaneously making authentication procedures as convenient as possible for both authorized persons and facility administrators. Such goals are frequently incompatible with each other.

The use of a password is perhaps the simplest and least expensive technique for providing access security. Additionally, passwords are relatively easy to change. However, there are problems with passwords; when they are fixed for long periods of time the chances of guessing them are improved; and when they are changed too frequently, they are forgotten by the rightful users. Further, when passwords are transmitted across an interface, they can be intercepted by anyone with the proper monitoring equipment.

In one known system, a common secret code is stored within each of two devices (key and lock). The secret codes are logically combined with a random number, available to each device, and the resulting numbers are compared with each other for identity. This technique is generally employed by various data communication systems (see e.g. "Locking Up System Security" - Electronics Week February 18, 1985 regarding Intel Corporation's 27916 KEPROM™ Keyed Access EPROM). Advantageously, the secret code itself needs never be transmitted so that an electronic intruder, monitoring interface signals, sees only the random data (challenge) and the modified random data (response) which are insufficient to teach the correct response to subsequent challenges. Unfortunately, this technique stores the same secret code in all keys which precludes selective revocation of lost or stolen keys.

One way to prevent tampering with private information in electronic systems is the use of cryptosystems (i.e., methods for encrypting, or transforming, information so that it is unintelligible and, therefore, useless to those who are not meant to have access to it). Ideally, the transformation of the information is so complicated that it is beyond the economic means of an eavesdropper to reverse the process. The eavesdropper is ther fore not inclined to become an intruder who not only would compromise the confidential nature of the stored information, but also might engage in forgery, vandalism and theft. A popular technique, known as public-key cryptography, relies on the use of two keys - one to encode the information and another to decode it. These keys are related in the sense that they serve to specify inverse transformations; however, it is computationally infeasible to derive one key from the other. That being the case, one of the keys can be made public for improved convenience without compromising the security of such a system. Applying public-key cryptography to the challenge of excluding unauthorized persons seeking entry to a secure facility, the party seeking entry would use his private key to encrypt (authenticate) a message. The party receiving the encrypted message would use the public key of the transmitter to decrypt the incoming message in order to transform it to its original text. A discussion of such systems is contained in the August, 1979 issue of Scientific American in an article by Martin E. Hellman entitled "The Mathematics of Public-Key Cryptography." An example of a public-key system is disclosed in U.S. Patent 4,453,074 issued to S. B. Weinstein for a "Protection System for Intelligent Cards." Unfortunately, in public-key systems, the party receiving the encrypted message must maintain a database that contains the public keys of all parties having authorization to enter the secure facility.

One particularly promising system involves the use of a password along with a smart card that exchanges data with an authentication device during an authentication procedure. It is noted that the smart card contains a processor and a memory; it is portable and frequently has the shape of a conventional credit card. Security is improved by requiring the holder of the smart card to remember a password. This password can either be sent to the smart card enabling it to exchange data with the authentication device, or the password can be sent directly to the authentication device itself. In either case, two conditions must now be satisfied: something in the user's head and something in the user's hand.

A known system stores an identification (ID) number within each smart card which is transmitted to the authentication device in order commence the authentication procedure. The authentication device scrutinizes the ID number to determine whether it corresponds to a presently valid ID number and then commences the authentication procedur oniy when the result is affirmative. Such a system is disclosed in U.S. Patent 4,471,216. While personal identification numbers additionally offer the ability

to improve flexibility (e.g., expiration date may be built into the ID itself), the storage of each individual ID number in the authentication device requires significant memory space. For example, storing 25,000 user keys, each 8 bytes long, requires 200K bytes of memory. Further, each time a new smart card is issued, the memory of the authentication device must be updated to recognize it. This is particularly impractical in a distributed system where, for example, the authentication device is used in connection with room or building access. Even when the authentication device comprises a host computer that is easily updated, it is undesirable from a security standpoint to store all ID numbers therein because they might be compromised if someone found a way to break into the computer.

Summary of the Invention

A security system includes a portable object, such as a smart card, and an authentication device for electrically interacting with the portable object to regulate access to a secure facility. An identification number $(ID)_n$ is presented to the authentication device which uses an encryption algorithm, E_1, to convert it into a secret code S_n. The authentication device also generates a challenge number, C, which is transmitted to the portable object. Stored within the portable object is secret code S_n and encryption algorithm E_2 which are used together with the challenge number C to create a response signal R_n. Stored within the authentication device is encryption algorithm E_2, which is used together with secret code Sn and the challenge number C to create response signal R_n. A favorable comparison between R_n and R'_n is necessary to gain access to the secure facility.

In an illustrative embodiment of the invention, E_1 and E_2 are identical processes that use different master strings (secret keys) to transform a first binary number into a second binary number. Knowledge of the encryption algorithm, however, is insufficient for an intruder to determine the master string. The present invention illustratively uses the Data Encryption Standard (DES) in the implementation of E_1 and E_2.

In a preferred embodiment of the invention, challenge number C is a 64-bit random number. Such numbers are generally non-repeating and enhance security by virtue of their non-predictable character.

The present invention advantageously regulates access to any one of a number of protected resources including information, cash, and physical entry into a facility without requiring the transmission of secret information across an interface. Importantly, the present invention eliminates the need to store and administer identification information regarding each user entitled to access to the protected resources.

It is a feature of the present invention that multiple secret codes are easily stored within a smart card, each providing access to a different facility, or backup access to the same facility in the event of a security breach (e.g., the master string becomes known). In the situation that security is breached, new secret codes can be derived at the authentication device by merely using a new master string. Such new secret codes would have already been stored within each smart card at the time of issue as a precautionary measure. Thus, should security become compromised, new smart cards do not need to be issued.

These and other features of the present invention will be more fully understood when reference is made to the detailed description and associated drawing.

Brief Description of the Drawing

FIG. 1 is a flow diagram illustrating the various steps performed in practicing the invention;

FIG. 2 is a flow diagram of the enciphering computation of the Data Encryption Standard;

FIG. 3 is a block diagram that illustrates the calculation of f(R,K) used in the Data Encryption Standard;

FIG. 4 discloses selection table S_1 used in the Data Encryption Standard;

FIG. 5 is a block diagram representation of the major functional components of a smart card system and their general interconnection with each other;

FIG. 6 illustrates use of the present invention in a computer access security system in accordance with the invention;

FIG. 7 illustrates use of the present invention in a premises access security system in accordance with the invention;

FIG. 8 discloses the functional components of a door lock such as used in connection with FIG. 7;

FIG. 9 illustrates the structure of a master string used in the encryption process;

FIG. 10 illustrates the structure of a challenge signal including information regarding the selection of the secret code to be used during the encryption process; and

FIG. 11 discloses a pseudo-random number generator suitable for use as a challenge number generator.

Detailed Description

GENERAL

Referring to FIG. 1, there is disclosed a diagram which illustrates the salient features of the invention in modified flow chart form. The mechanical analog of a key and a lock is useful in connection with FIG. 1 because smart card 500 functions as a key and authentication device 700 functions as a lock. Since the authentication process requires activity on the part of both the smart card and the authentication device, the activity associated with each part is segregated to assist the reader in understanding the invention. Although not requned in the practice of the invention, security is enhanced by requiring the holder of the smart card to enter a password into the smart card, enabling it to commence the authentication process by transmitting a personal identification number $(ID)_n$ to authentication device 700. Alternatively, the holder of the smart card could directly transmit $(ID)_n$ to the authentication device 700. In either case, the following steps describe the authentication process: (1) In response to the receipt of a signal such as $(ID)_n$, box 740 recognizes the signal and initiates the generation of a challenge number. Additionally, secret code S_n is created (box 710) using encryption algorithm E_1 (box 730) and the proffered personal identification number $(ID)_n$. (2) Challenge number C is generated (box 750), transmitted to smart card 500, and used internally (box 720). Note that a valid ID number is not required to initiate the generation of a challenge number - a feature that helps preserve confidentiality of the ID number. (3) Both the smart card 500 and the authentication device 700 (box 563 and box 720) calculate a response (R_n and R'_n respectively) to the challenge number. Since secret code S_n and encryption algorithm E_2 are contained in both the smart card and in the authentication device, the responses should be identical when compared (box 760). (4) Block 770 further enhances security, with minimum inconvenience to the system administrator, by testing whether the proffered $(ID)_n$ corresponds to a lost or stolen card. The list of such cards is presumably small and is seldom updated. Once all of the above steps have been successfully completed, access to the computer is granted, a door is opened, a credit transaction is validated, or cash is delivered, etc.

The various boxes need not reside within the particular device as shown in FIG. 1. For example, in a number of applications, the challenge number generator can be located within the smart card while still preserving the benefits of th invention. Indeed, in the peer-to-peer authentication application described hereinafter, each smart card contains a challenge number generator, means for comparing response numbers, and the E_1 algorithm including a master string. Further, user interface 100 can be built into the smart card 500 or the authentication device 700. It is an important advantage that the list of valid ID numbers need not be stored within the authentication device. It is sufficient that only the encryption algorithm E_1, originally used to create S_n from $(ID)_n$, needs to be stored.

Stored within memory box 550 of smart card 500 is the above-identified personal identification number $(ID)_n$ that is unique to that card. Also stored within box 550 are one or more secret codes S_n and encryption algorithm E_2.

Secret code S_n comprises a plurality of binary digits stored in memory that are not accessible from outside the card. Further, S_n is written into memory at a time when the ID number is first assigned by the card issuer. S_n is linked to a particular personal identification number, designated, $(ID)_n$ by the functional relationship $S_n = E_1 - (ID)_n$. What this means is that encryption algorithm E_1 maps each unique personal identification number into a unique secret code. As a practical matter, a secret computer program transforms input signal $(ID)_n$ into output signal S_n. It is the use of this particular transformation that eliminates the need to store individual ID numbers. More will be said about this later.

Encryption algorithm E_2 is a computer program executed by a microprocessor. It is jointly responsive to secret code S_n and to input binary data signal C for generating an output binary data signal R_n. Computation of R_n is indicated in box 563 where C is the challenge number and R_n is the response. For improved security, C is a large nonrepeating number so that an intruder making a large humber of observations of the challenge and response will never learn the manner by which they are related. So long as C and S_n are finite, however, it is theoretically possible for the determined intruder to learn the correct response to all challenges. Nevertheless, with a moderate length secret code, say 64 bits, there are approximately 18 x 10^{18} possible unique secret code combinations. Even with a computer aided lockpick that tried 10 billion different combinations every second, it would take 57 years to examine all combinations. This period could be lengthened substantially if additional delay, say 1 second, was introduced between challenge and response. By way of example, and not limitation, C may be a random number, pseudo-random number, or even a time clock (year: month: day: hour: seconds: tenths: etc.).

Stored in box 770 are the ID numbers of lost and stolen cards as well as numbers that have expired or, for one reason or another, no longer have permission to access the facility. Advantageously, even though the authentication device

"knows" at the outset that the proffered ID number is unacceptable, access to the facility is not denied until the entire process has been completed. Thus, only minimum information is given to potential intruders. Storing a list of unacceptable numbers allows customization with minimum susceptibility to fraud. There is little or no incentive to increase the list of unacceptable ID numbers; while on the other hand, a great temptation exists to fraudulently increase the list of acceptable ID numbers - a temptation that the present invention eliminates.

DATA ENCRYPTION STANDARD (DES)

The purpose of any encryption algorithm is to convert confidential information (data) into a form that renders it unreadable to all except those who know how to decode the message. One simple technique involves substituting one letter of the alphabet with another for each of the letters. Such encryptions, however, are relatively easy to decrypt, even for the unsophisticated intruder. More complex techniques have arisen over the years to stay ahead of unsolicited decryption experts, and the art has progressed to the point that techniques exist that are so good that it no longer makes sense to try to unravel an encryption signal. One such technique that has gained wide acceptance is the Data Encryption Standard (DES) that is intended for implementation in special purpose electronic devices. In 1977, the National Bureau of Standards (now NIST) issued DES as a Federal standard, and the National Security Agency has certified new products using the standard. While a relatively brief discussion of the application of DES to the invention is set forth below, a more comprehensive treatment is set forth in the January 15,1977 Federal Information Processing Standards Publication 46 (FIPS 46), entitled "Specifications for the Data Encryption Standard."

DES is a private-key scheme in which both encrypting and decrypting keys are identical and secret. DES operates on data in blocks of 64-bits, sending it through 16 stages of the algorithm before exiting as a 64-bit cipher text. Encryption relies heavily on proper management of keys - the strings of characters that must be input to the algorithms before encryption or decryption can take place. The present invention does not require decryption, but rather relies on a comparison between two encrypted signals. Encryption algorithms E_1 and E_2 each use DES to achieve encryption; however, the data blocks and keys are obtained from different sources. After a brief explanation of DES is given, it will be applied to the present invention.

A flow diagram that illustrates the sequential operations performed in the DES enciphering computation is shown in FIG. 2. Input box 201 comprises a 64-bit ordered set (vector) of binary digits whose order is rearranged (permuted) according to a known pattern in an operation akin to shuffling cards. The permuted block of 64-bits is now split into two boxes 203 (L_0) and 204 (R_0), each comprising 32-bits in an operation akin to cutting the cards. At this point, the card shuffling analogy fails because mathematical operations 205 (modulo-2 addition) and 206 (cipher function f) are introduced along with key K. Values for $K_1 \ldots K_{16}$ are selected in accordance with 16 different predetermined schedules whereby each K_n comprises an ordered set of 48-bits chosen from the 64-bit key.

For completeness, the operation of cipher function (f) is shown in FIG. 3 where the calculation f(R, K) is diagrammatically laid out. In this figure, E denotes a function which takes a block of 32-bits as input and yields a block of 48-bits as output. The E function is very similar to the initial permutation of box 202, but now certain of the bits are used more than once. These blocks of 48 bits, designated 303 and 304 in FIG. 3, are combined by modulo-2 (exclusive or) addition in box 305. Selection functions $S_1, S_2, \ldots\ldots S_8$ take a 6-bit input number and deliver a 4-bit output number in accordance with a predetermined selection table such as shown in FIG. 4 which discloses the S_1, function. For example, if S_i is the function defined in this table and B is a block of 6 bits, then $S_1(B)$ is determined as follows: The first and last bits of B represent, in base 2, a number in the range 0 to 3. Let that number be i. The middle 4 bits of B represent, in base 2, a number in the range 0 to 15. Let that number be j. Look up in the table the number in the i^1th row and j^1th column. It is a number in the range 0 to 15 and is uniquely represented by a 4-bit block. That block is the output $S_1(B)$ of S_i for the input B. Thus, for input 011011 the row is 01 (i.e., row 1) and the column is determined by 1101 (i.e., column 13). In row 1, column 13 the number 5 appears so that the output is 0101. Selection functions, $S_1, S_2, \ldots S_8$ appear in the Appendix of the above-mentioned publication FIPS 46.

Referring once again to FIG. 3, the permutation function P is designated 306 and yields a 32-bit output (307) from a 32-bit input by permuting the bits of the input block in accordance with table P, also set forth in FIPS 46.

ENCRYPTION ALGORITHMS E_1 AND E_2

DES is now applied to encryption algorithm E_1

which is used to convert $(ID)_n$ to S_n. Note that when the smart card is issued, it comes equipped with S_n already stored in its memory. Reference is now made to FIG. 9 which illustrates the structure of the master string which comprises 640-bits of secret data used by the encryption algorithm E_1. The master string is interpreted as 10 separate characters (addressable by digits 0-9), each having 64 bits of data. The ID number comprises a block of 6 digits, each assuming some value between 0 and 9 inclusive. In the following example, encryption algorithm E_1 operates on $(ID)_n$ (illustratively set equal to 327438) in the manner indicated. The first operation requires that the third character of the master string be combined with the second character of the master string in accordance with the DES enciphering computation. This operation is denoted d(3,2) where 3 is treated as the data block and 2 is treated as the key. The operation performed is shown in FIG. 2 in which the 64-bit number corresponding to the third character of the master string is used as input 201, the 64-bit number corresponding to the second character of the master string is used as K, and output 210 is a 64-bit number (designated "A") that will be used in a second operation.

The second operation performed is similar to the first except that "A" is combined with the seventh character of the master string in accordance with the DES enciphering computation. This operation is denoted by d(A,7) where A is a 64-bit number used as input 201, and the 64-bit number corresponding to the seventh character of the master string is used as K. The operation performed is shown in FIG. 2 and output 210 is a 64-bit number (designated "B") that will be used in a third operation.

These operations continue until all of the digits of $(ID)_n$ are used. The last operation, d(D,8), results in a 64-bit number which is used as the secret code S_n. Accordingly, in this example, encryption algorithm E_1 uses the digits of $(ID)_n$ to index characters of the master string. The DES enciphering computation shuffles these secret keys in a known, but non-reversible, manner to generate S_n.

DES is now applied to encryption algorithm E_2 which is used to convert S_n and C into a response number R_n (within the smart card), or R'_n (within the authentication device). S_n and C each comprise a 64-bit number which makes them ideally suited for the encryption computation shown in FIG. 2. Indeed, S_n and C are "shuffled" in accordance with the DES enciphering computation described above (see FIG. 2), and output box 210 now contains a 64-bit number designated R_n or R'_n. These numbers are thereafter compared, and when they are identical the smart card is deemed to be authenticated. Although the DES enciphering computation is illustratively shown, it is understood that other enciphering computations, having greater or lesser complexity, may be used without departing from the spirit of the invention.

CHALLENGE NUMBER GENERATOR

There are many techniques for generating suitable challenge numbers. Ideally such numbers are long, non-predictable, non-repeating and random. One known technique involves periodically sampling the polarity of a noise source, such as an avalanche diode, whose average dc output voltage is zero. As discussed above, the challenge number generator 750 (FIG. 1) may generate a random number, a pseudo-random number, or even a predictable number - depending on the degree of security warranted in the given application. One challenge number generator is shown in FIG. 11 which provides a pseudo-random number at its serial data output The generator comprises a 64-stage shift register whose output is modulo-2 combined (via Exclusive-OR gates 111,112) with various of its stages and then fed back to the input of the generator. Although the serial data output pattern is very long (potentially generating all possible combinations of 64 bits), it eventually repeats itself. Nevertheless, by accelerating the clock rate at times when a challenge number is not needed, it would be most difficult to predict which particular combination of 64 bits was coming next.

The randomness of the challenge number is further improved by using the DES enciphering computation shown in FIG. 2. Here, the Parallel Data Output (X_0, ... X_{63}) of the pseudo-random number generator shown in FIG. 11 is used as input 201 in FIG. 2, while one character of the secret master string is used in obtaining the various values for K. Recall that values for K_1... K_{16} are selected in accordance with 16 different predetermined schedules whereby each K_n comprises an ordered set of 48-bits chosen from a 64-bit key. Since the software needed to implement DES, or the particular encryption algorithm used, is already in place in both the smart card and in the authentication device, it is cost effective to use it in connection with the generation of a challenge number. Indeed, if DES is used in forming the challenge number, it would be sufficient to increment a register each time a new challenge number is needed, and then use that number, rather than X_0,... X_{63}, as input 201 in FIG. 2.

SMART CARD

Referring now to FIG. 5 there is disclosed a block diagram of a smart card 500 and a reader/writer unit 900 such as used in connection with the present invention. Although shown in greater detail in U.S. Patent 4,798,322, a brief description is presented here. Some of the principal components located on smart card 500 are microprocessor 560, electrically erasable programmable read-only memory (EEPROM) 550, analog interface circuit 540, secondary winding 521 of transformer 920, and capacitive plates 541-544.

Microprocessor 560 includes a central processing unit and memory means in the form of random access memory and read-only memory. A microprocessor available from Intel Corporation such as Part No. 80C51 may be used with the proper programming. Operating under firmware control provided by its internal read-oniy memory, the microprocessor 560 formats data to the EEPROM 550 and to the reader/writer unit 900 via the analog interface circuit 540. EEPROMS are available from a number of suppliers, many of whom are mentioned in an article entitled "Are EEPROMS Finally Ready to Take Off" by J. Robert Lineback, Electronics, Vol 59, No. 7, (Feb 17,1986), pp. 40-41. Data may be written to or used from an EEPROM repeatedly while operating power is being applied. When operating power is removed, any changes made to the data in the EEPROM remain and are retrievable whenever the smart card 500 is again powered.

The analog interface circuit 540 provides a means for interfacing smart card 500 with reader/writer unit 900. Within analog interface 540 are circuits responsive to capacitors 541-544, for exchanging data with reader/writer unit 900. Power for operating the card 500 is provided to the analog interface circuit 540 via inductive transfer, received by the secondary winding 521 of transformer 920. This transformer is formed when secondary winding 521 is coupled to a primary winding 921 within the reader/writer unit 900. The transformer 920 may advantageously include a ferrite core 922 in the reader/writer for increased coupling between the transformer primary winding 921 and secondary winding 521. A second such core 522 may also be included in the transformer 920 to further increase coupling efficiency. The primary winding 921 is driven at a 1.8432 MHz rate by power supply 930 whose operation is described with particularity in U.S. Patent 4,802,080 issued January 31,1989.

Within the reader/writer unit 900, analog interface circuit 940 exchanges data with the smart card 500 under control of microprocessor 960. Capacitor plates 941-944 ar aligned with the mating capacitor plates 541-544 within the smart card 500. The input/output serial data interface 950 is basically a universal asynchronous receiver transmitter (UART) which may be advantageously included in the microprocessor 960. This UART is used for externally communicating with a suitably configured application station 990.

Application station 990 represents any one of a variety of stations, terminals or machines capable of interacting with the reader/writer unit 900 for the purpose of selectively granting access to the resources which it controls such as cash, premises access, information in a computer, credit authorization for a telephone call or the purchase of goods, etc. Stored within the application station is the computational power to carry out the authentication procedure disclosed in FIG. 1. Reader/writer unit 900 may itself be part of the application station 990 and its microprocessor 960, when provided with sufficient memory, is suited to carry out the authentication procedure. Also stored within the application station is the appropriate hardware to open a lock or remit cash. Such hardware is well known by those in the particular art to which the application station pertains. A discussion of certain of these applications follows.

APPLICATIONS

Computer Access Security System

FIG. 6 discloses one application of the present invention in a computer access security system. In this system, terminal stations 101 and 102 provide access to host computer 600 so long as the user can be authenticated. In one situation, the user inserts his smart card 501 into a terminal security server (TSS) 610 for the purpose of verifying that he is entitled to access host computer 600. Modems 641 and 643 are frequently needed to adapt digital signals to transmission over public switched network 650. At the host location, host security server (HSS)630, together with host smart card 503, grants access only to authorized users. In this application, TSS 610 includes a reader/writer unit 900 such as shown in FIG. 5, that interacts with smart card 501 to exchange electrical signals between the smart card and a particular application station. The user transmits his password to smart card 501 via terminal station 101 which commences the authentication process with HSS 630 and host smart card 503. Security is improved by storing the authentication algorithms and master strings within smart card 503 rather than in the host computer. Whereas a super-user might be able to access secret codes stored within the host computer 600, the host smart card is configured to only

grant or deny access; secret information within the card 503 is not available to anyone after it has been entered. Since individual user ID numbers do not have to be stored in the present invention, it is possible to handle the authentication of vast numbers of users with minimal storage so that smart cards using EEPROMS of moderate size, say 2048 bytes, are adequate for the task. The authentication process performed in this application is the same as discussed above using DES or another suitable enciphering computation.

Variations of this system include the situation where the TSS 610 is replaced by a portable security server (PSS) 620. Here, the user types his identification number $(ID)_n$ into terminal station 102. $(ID)_n$ is then transmitted to HSS 630 which includes host smart card 503. HSS 630 returns a challenge number which is displayed on terminal station 102. The user then enters this challenge number into PSS 620 using keys 622. Contained within PSS 620 is smart card 502 which stores secret code S_n and encryption algorithm E_2. It computes a response R_n to the challenge number and displays it on liquid crystal display 621. Thereafter, the user enters R_n into terminal station 102 and awaits access to host computer 600. Clearly, each terminal station 101,102 could contain the equipment presently housed within TSS 610 or PSS 620.

Premises Access Security System

An important application of the present invention is in connection with the replacement of conventional door locks and mechanical keys where high security is important. Smart cards are useful in this application because they can be selectively revoked and adapted for use only during predetermined hours. Further, they can be programmed to commence or expire on certain dates. The present invention is particularly advantageous in such a distributed system because the identity of each newly authorized user does not have to be communicated to each lock, although information regarding users no longer having authorization must be so communicated. The security of microwave "huts," which control vital junction points in the national telecommunication network, is of critical importance. Such locations warrant greater protection than easily duplicated mechanical keys can offer.

An example of a premises access security system is shown in FIG. 7 which illustrates another application of the present invention. Door 830 provides entry to a secure location such as a room or a building. Outside handle 850 does not normally operate the lock, but is provided merely for conveniently pushing or pulling on the door once the lock is open. A bolt assembly is driven by an inside handle (not shown) and includes a protrusion 840 which engages a strike 995 positioned in the door jamb. In the embodiment of FIG. 7, the strike itself is activated to permit the opening and closing of the door. Alternatively, the bolt within the door could have been controlled in accordance with the invention. Lock 800 is positioned adjacent the door jamb on wall 820 and includes a slot 810 for inserting an electronic key.

Referring now to FIG. 8, additional detail is provided regarding the hardware needed to support this particular application. In order to obtain access, the user first inserts his key 500 (smart card) into slot 810 (see FIG. 7) of lock 800. Once the key 500 is in contact with reader/writer unit 900, as discussed in connection with FIG. 5, authentication can begin. The user enters his password using the switches 120 on user interface 100 which is transferred to key 500 via reader/writer unit 900. If the entered password matches the password stored in memory 550 of key 500, then the key transmits its identification number $(ID)_n$ to application station 990, and more particularly to authentication device 700 which carries out the authentication procedure discussed in connection with FIG. 1. In the event that the key is authenticated, processor 760 delivers a pulse to relay driver 770 which activates relay 780 thereby closing contact K1. Power is now applied to electric strike 995 which enables the door to be pulled open. A suitable transducer for carrying out this function is the Model 712 Electric Strike, manufactured by Folger Adam Co. that requires 12 volts DC at 0.3 amperes. Information regarding door entry may be delivered to the user on display 110 of the user interface 100. Such information might include prompts for using the system, a message that the key has expired or that the password should be re-entered. Processor 760 includes memory for storing encryption algorithms E_1 and E_2 as well as a list of lost/stolen keys and those ID numbers that have been granted access to the facility over some time period. Such information can be delivered to, and displayed on, user interface 100 when properly commanded.

Multiple Secret Codes

In accordance with the present invention, the smart card may be used in connection with a plurality of authentication devices in which each device grants access to different user population. This is made possible by storing a plurality of secret codes within each smart card - very much like having a number of different keys on a single key ring. Knowing which s cret code to use is communicated to the smart card when the chal-

lenge is delivered. Recall that challenge C comprises a 64-bit (8 byte) random number in the preferred embodiment. An additional byte (header) is added to the challenge, as shown in FIG. 10, that selects one of the secret codes S_n stored within the memory of the smart card. Here, the header corresponds to the address of the particular secret code to be used in providing the correct response to the challenge. An 8-bit header accommodates 256 different secret codes, many of which may be used to enhance the security of a single authentication device. Perhaps 2 or 3 different challenges might be issued in an extremely high security application. In situations where 64-bits of random data are not necessary, various bit positions of the challenge number can be dedicated to identifying the particular secret code to be used.

Peer-to-Peer Authentication

In a number of situations, it is desirable for authentication to proceed between two members of a population who desire to exchange secret iniormation after the identity of each member is verified to the satisfaction of the other. The present invention is useful in this regard because it does not require storage of the identification numbers of all members of the population. However, each of the smart cards must generate a challenge signal, store secret code S_n as well as encryption algorithms E_1 and E_2, and compare response numbers with R_n with R'_n. Authentication proceeds in a manner similar to the procedure of FIG. 1, except that the combined functions of smart card 500 and authentication device 700 are now contained within a single, more powerful smart card. After the first smart card authenticates itself to the second, the second smart card authenticates itself to the first. This assures the first user that he has reached the correct destination, and it assures the second user that the person seeking access is entitled to it. Since each smart card now carries the secret master string, security is potentially weakened. However, the master string is not retrievable from memory and cannot be determined by trial and error within a reasonable time.

Modifications and variations of the present invention are possible and include, but are not limited to, the following: (i) smart cards are portable devices that may assume any convenient shape; (ii) smart cards may include metallic contacts although the disclosed contactless interface offers great resistance to external contaminants and electrical discharge; (iii) challenge numbers need not be random or even secret, although some degradation to security is inevitable; and (iv) encryption algorithms E_1 and E_2 may be less complex than DES and may even be implemented in hardware comprising no more than an Exclusive-OR gate.

Claims

1. A system for controlling access to a secure facility, the system including a portable object (500) and means for transferring data between the portable object and the facility,
the facility comprising:
memory means for storing encryption algorithms E_1 and E_2;
means (750) for generating a challenge number (C);
means responsive to an identification signal (ID)-subn that identifies the particular portable object (500) seeking to gain access to the facility, and to encryption algorithm E_1 for generating a secret code (S_n);
means (720) responsive to the challenge number (C), to the secret code (S_n) and to encryption algorithm E_2 for generating a first response signal (R'_n); means (760) for comparing the first response signal (R'_n) with a second response signal (R_n) generated by the portable object, and for providing an enabling signal when the comparison is favorable;
the portable object (500) comprising:
memory means (550) for storing the secret code (S_n)and the encryption algorithm E_2; and
means (563) responsive to the secret code (S_n), to the challenge number (C) received from the facility, and to encryption algorithm E_2 for generating a second response signal (R_n) and transmitting same to the facility.

2. The system of claim 1 wherein the facility further includes:
means for storing a list of identification numbers not entitled to access the secure facility; and
means (770) for determining correspondence between the stored list of identification numbers and the identification signal that identifies the particular portable object seeking access to the facility, and for denying access to the facility when such correspondence exists.

3. The system of claim 1 wherein the means (710) for generating the secret code (S_n) comprises a first processor, jointly responsive to the identification signal and to a secret master string, for executing a predetermined sequence of steps in accordance with encryption algorithm E_1.

4. The system of claim 1 wherein the means (720) for generating the first response signal comprises a first processor, jointly responsive to the secret code (S_n) and to the challenge number (C), for executing a predetermined sequence of steps in accordance with encryption algorithm E_2.

5. The system of claim 1 where... the means (563) for generating the second response signal comprises a second processor, responsive to the secret code and to the challenge number, for executing a predetermined sequence of steps in accordance with encryption algorithm E_2.

6. The system of claim 3 wherein encryption algorithm E_1 is a process for encrypting data in accordance with the Data Encryption Standard.

7. The system of claim 5 wherein encryption algorithm E_2 is a process for encrypting data in accordance with the Data Encryption Standard.

8. The system of claim 1 wherein the challenge number is substantially random.

9. A method for testing the authenticity of a portable electronic device (500) and for enabling access to a secure facility when the portable electronic device is authentic, the method comprising the steps of:

storing encryption algorithms E_1 and E_2;

receiving an identification signal $(ID)_n$ that identifies the particular portable electronic device seeking access to the facility;

generating a secret code (S_n) in accordance with encryption algorithm E_1 using the identification signal as an input;

generating a challenge number (C) and transmitting same to the portable electronic device;

generating a first response signal (R'_n) in accordance with encryption algorithm E_2 using the secret code and the challenge number as inputs;

comparing the first response signal (R'_n) with a second response signal (R_n) generated by the portable electronic device; and

enabling access to the secure facility when the comparison is favorable.

10. The method of claim 9 further including the steps of:

storing a list of identification numbers not entitled to access the facility; and

denying access to the facility when the received identification signal corresponds to a identification number stored on the list of those not entitled to such access.

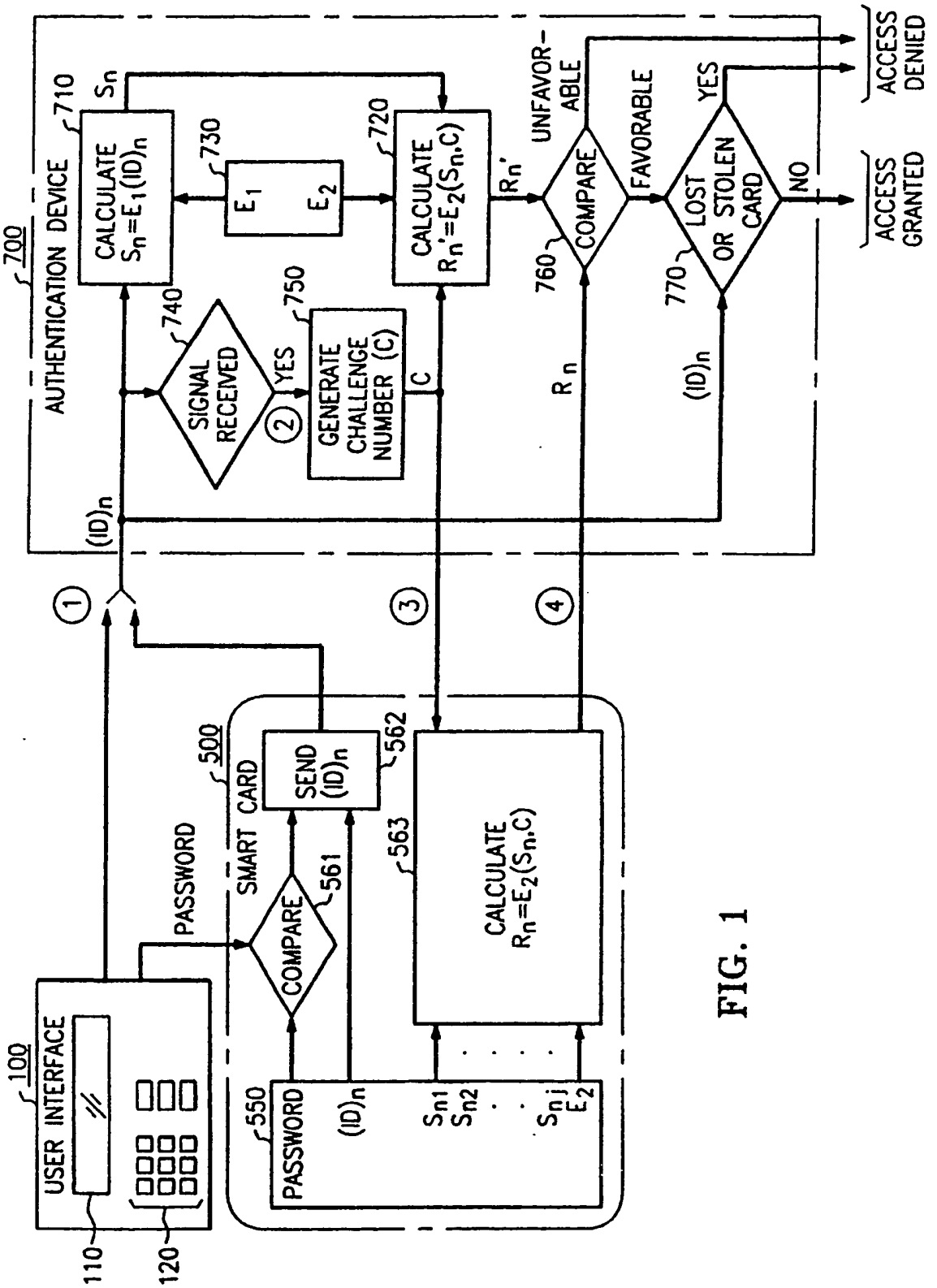
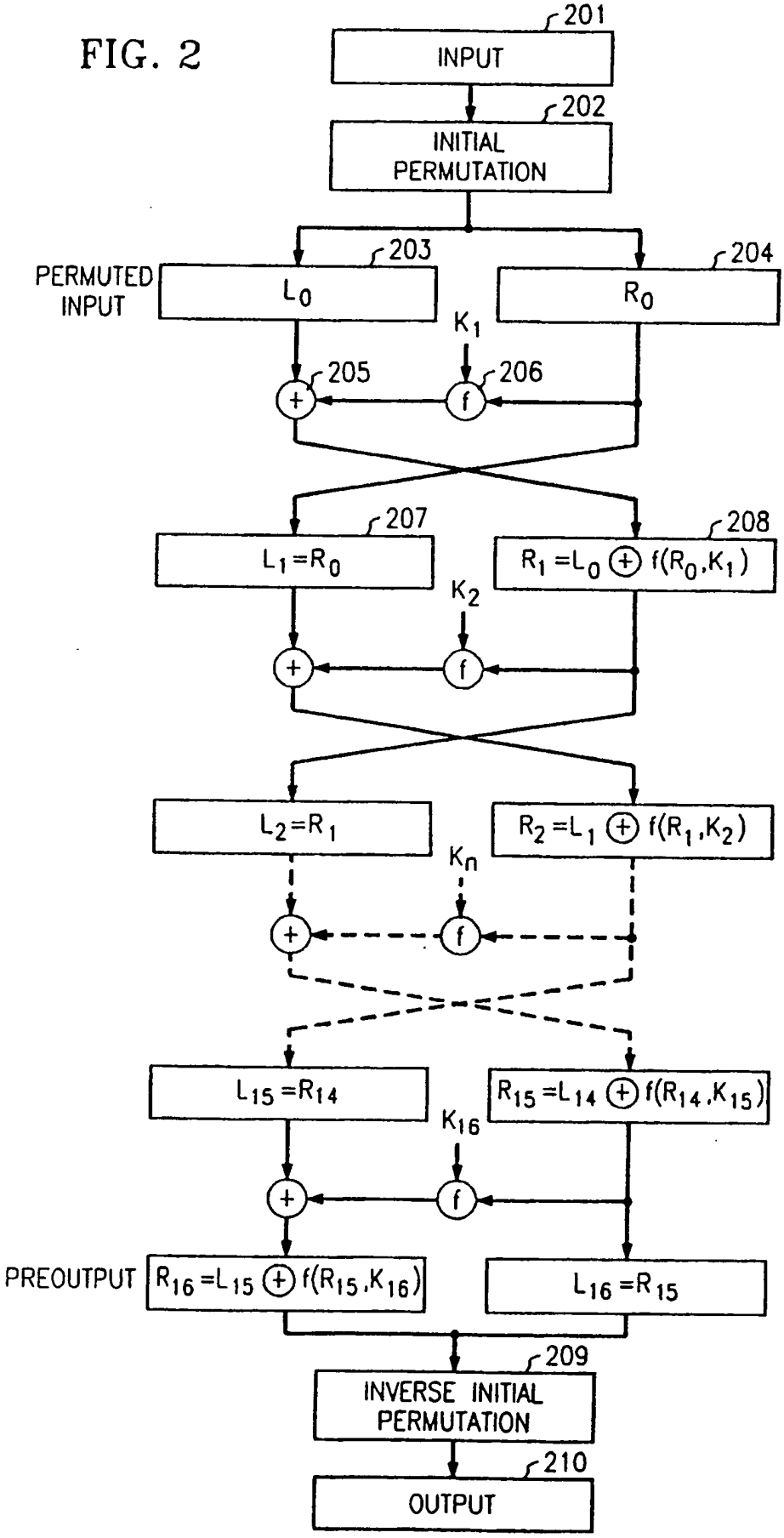

AUTHENTICATION DEVICE
700
CALCULATE $S_n=E_1(ID)_n$
710
S_n
730
E_1
E_2
720
CALCULATE $R_n'=E_2(S_n,C)$
R_n'
UNFAVOR-ABLE
COMPARE
760
FAVORABLE
LOST OR STOLEN CARD
770
YES
NO
ACCESS DENIED
ACCESS GRANTED
SIGNAL RECEIVED
740
YES
2
GENERATE CHALLENGE NUMBER (C)
750
C
R_n
$(ID)_n$
1
3
4
SMART CARD
500
SEND $(ID)_n$
562
563
CALCULATE $R_n=E_2(S_n,C)$
PASSWORD
COMPARE
561
USER INTERFACE
100
110
120
550
PASSWORD
$(ID)_n$
S_{n1} S_{n2} . . . S_{nj} E_2

FIG. 1

FIG. 2


201
INPUT
202
INITIAL PERMUTATION
PERMUTED INPUT
203
L_0
204
R_0
K_1
205
206
f
207
$L_1=R_0$
208
$R_1=L_0 \oplus f(R_0,K_1)$
K_2
$L_2=R_1$
$R_2=L_1 \oplus f(R_1,K_2)$
K_n
$L_{15}=R_{14}$
$R_{15}=L_{14} \oplus f(R_{14},K_{15})$
K_{16}
PREOUTPUT
$R_{16}=L_{15} \oplus f(R_{15},K_{16})$
$L_{16}=R_{15}$
209
INVERSE INITIAL PERMUTATION
210
OUTPUT

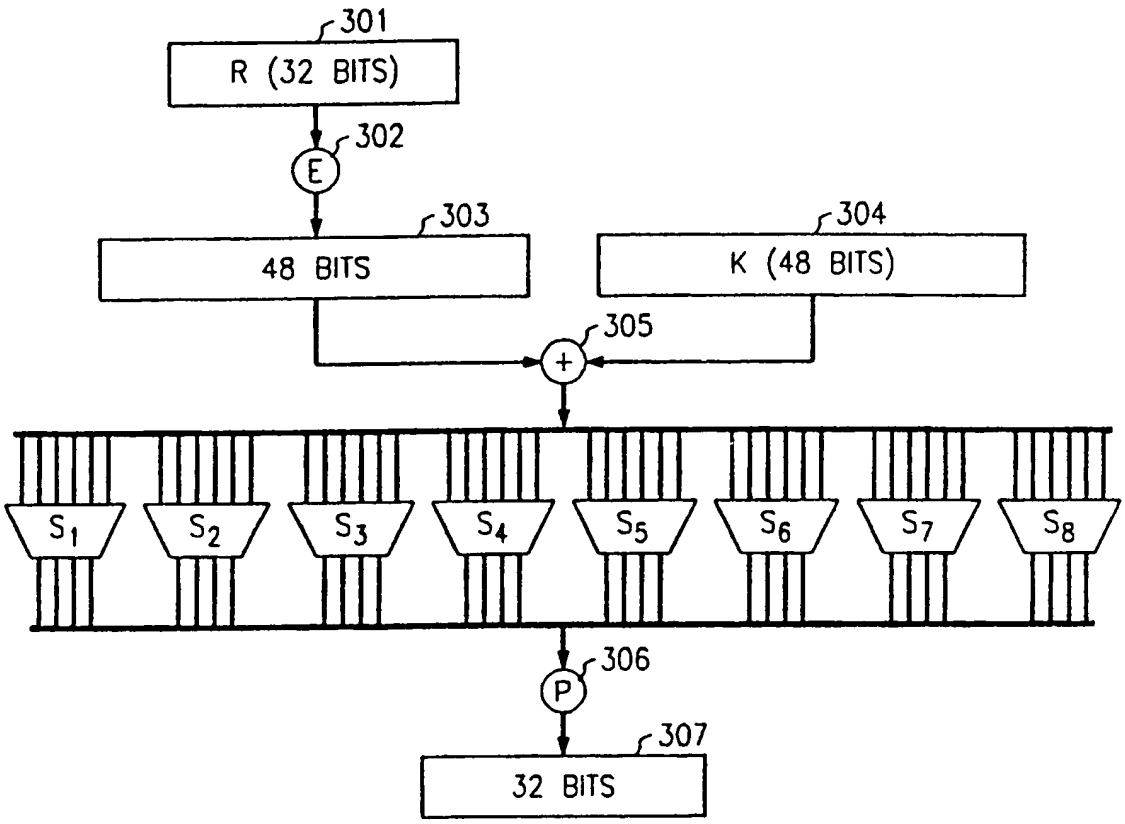


FIG. 3

		COLUMN No.															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ROW No.	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	5	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

FIG. 4

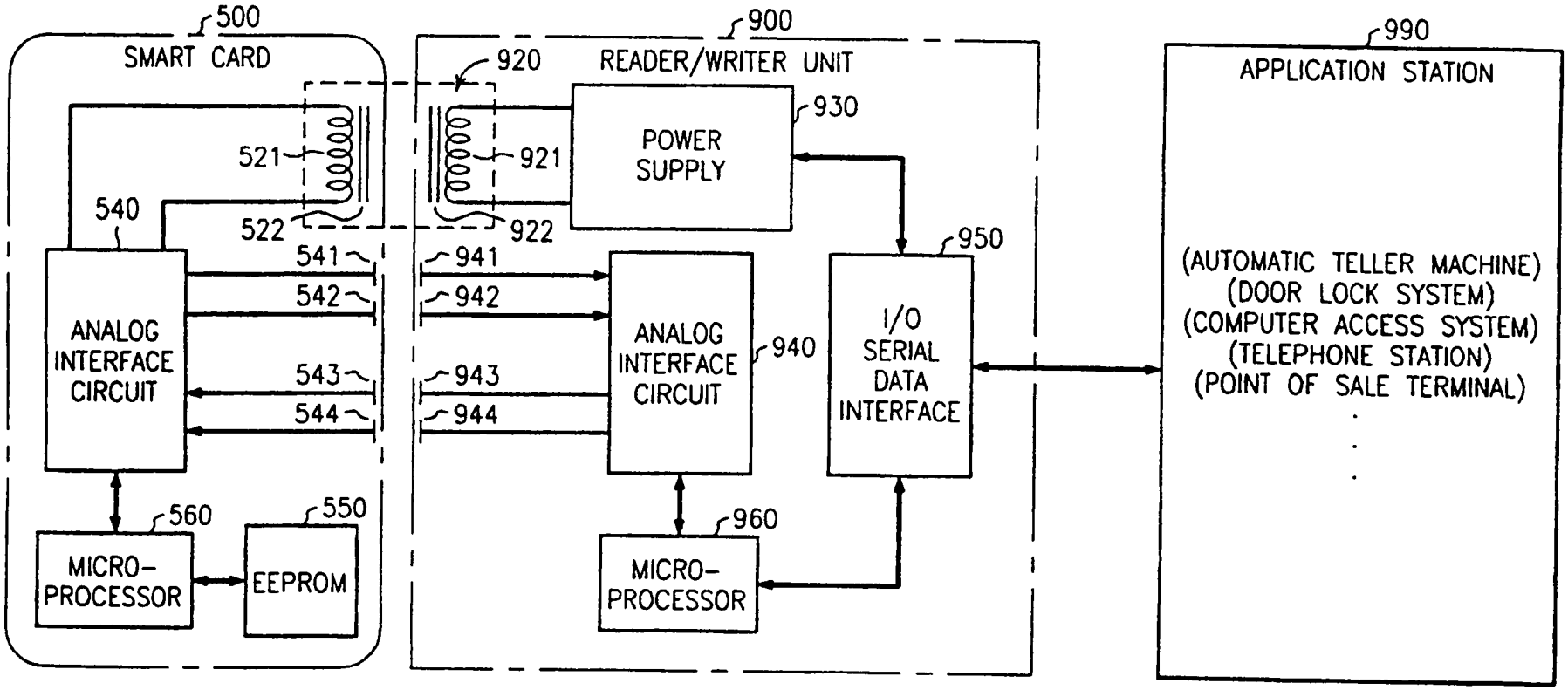

500
SMART CARD
900
READER/WRITER UNIT
920
990
APPLICATION STATION
521
921
930
POWER SUPPLY
540
522
922
950
541
941
542
942
ANALOG INTERFACE CIRCUIT
940
I/O SERIAL DATA INTERFACE
543
943
544
944
(AUTOMATIC TELLER MACHINE)
(DOOR LOCK SYSTEM)
(COMPUTER ACCESS SYSTEM)
(TELEPHONE STATION)
(POINT OF SALE TERMINAL)
560
550
960
MICRO-PROCESSOR
EEPROM
MICRO-PROCESSOR

FIG. 5

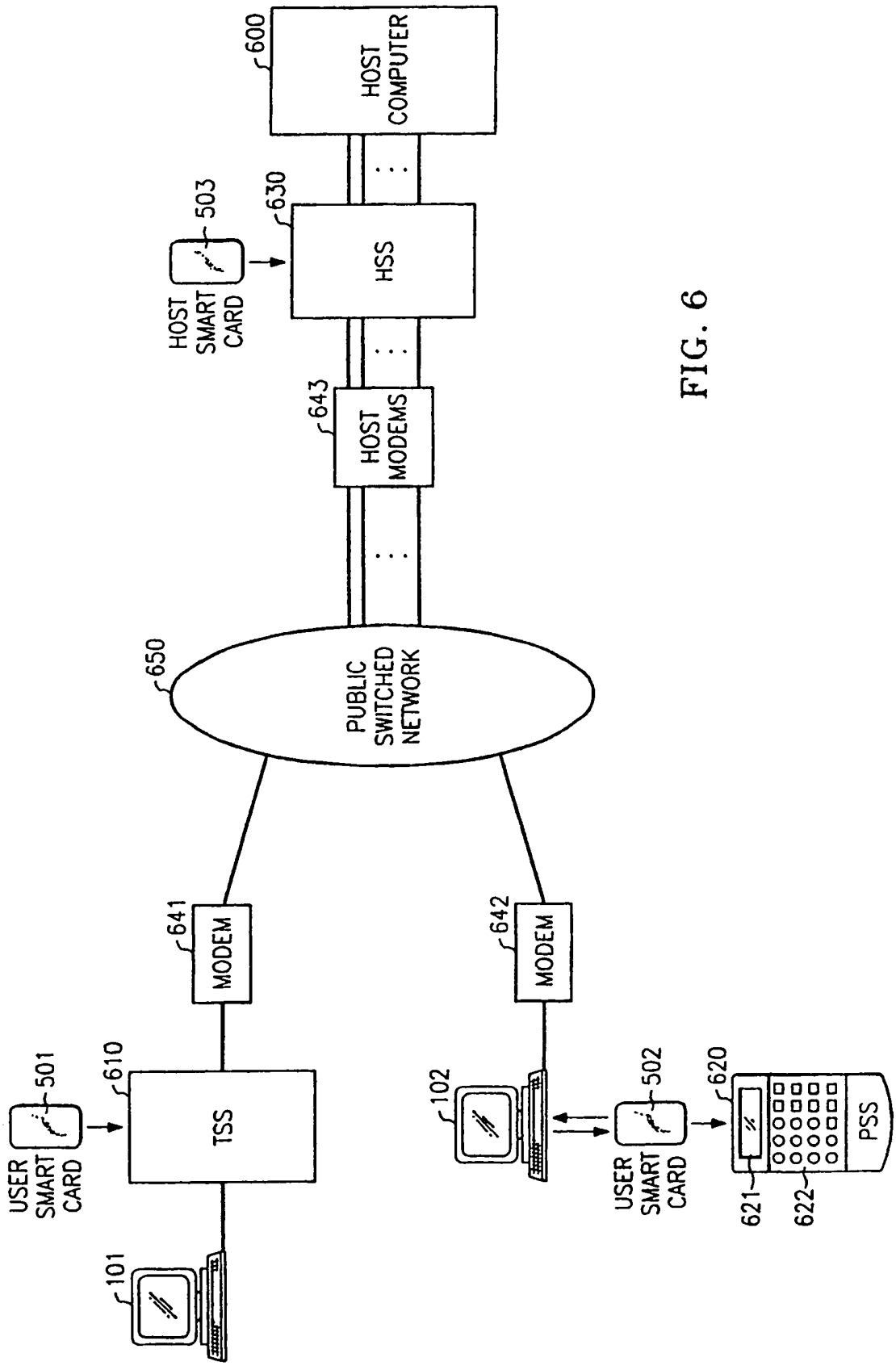

600
HOST COMPUTER
630
HSS
503
HOST SMART CARD
643
HOST MODEMS
650
PUBLIC SWITCHED NETWORK
641
MODEM
642
MODEM
501
USER SMART CARD
610
TSS
101
102
502
USER SMART CARD
620
PSS
621
622

FIG. 6

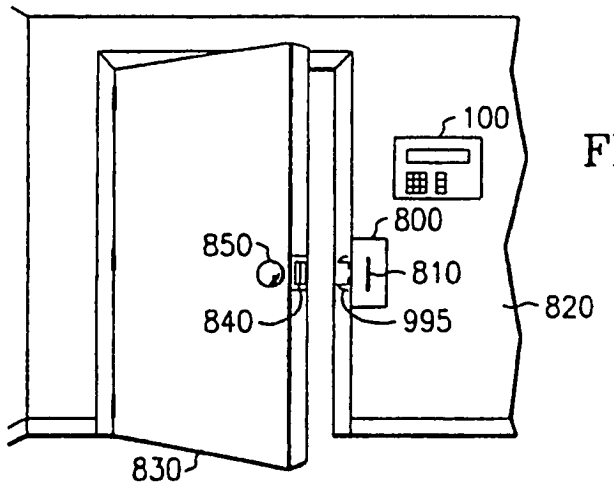
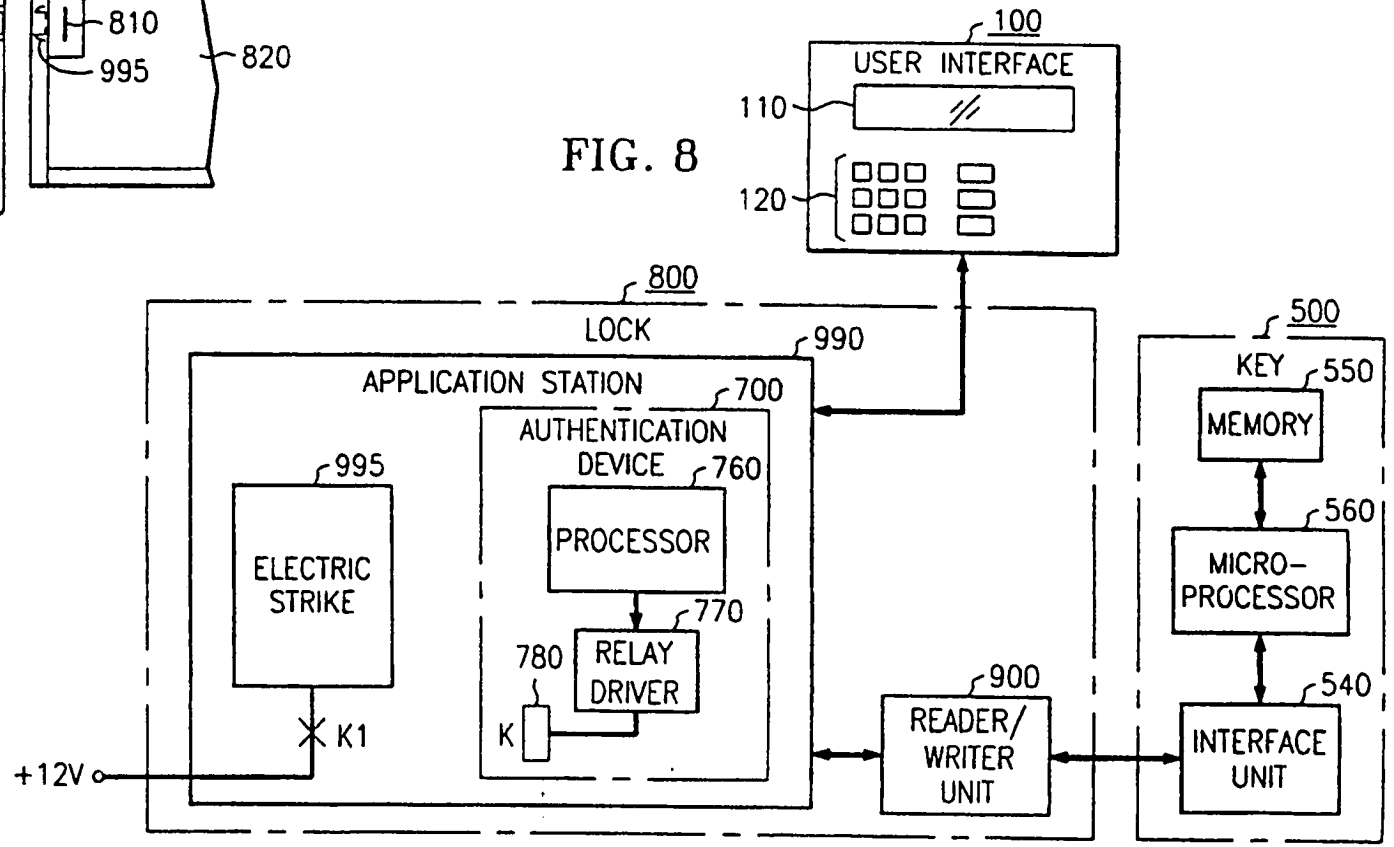

100
800
850
810
840
995
820
830

FIG. 7

FIG. 8


100
USER INTERFACE
110
120
800
LOCK
990
APPLICATION STATION
700
AUTHENTICATION DEVICE
760
995
ELECTRIC STRIKE
PROCESSOR
770
780
RELAY DRIVER
K
K1
+12V
900
READER/ WRITER UNIT
500
KEY
550
MEMORY
560
MICRO-PROCESSOR
540
INTERFACE UNIT

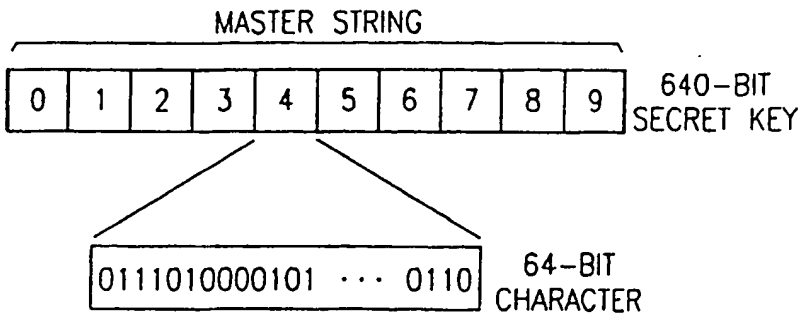


FIG. 9

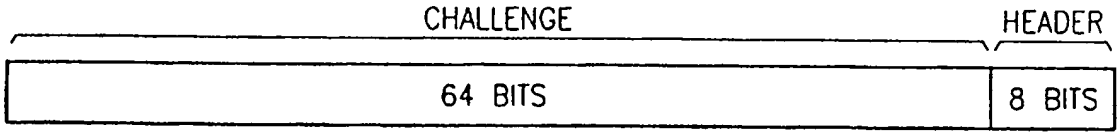


FIG. 10

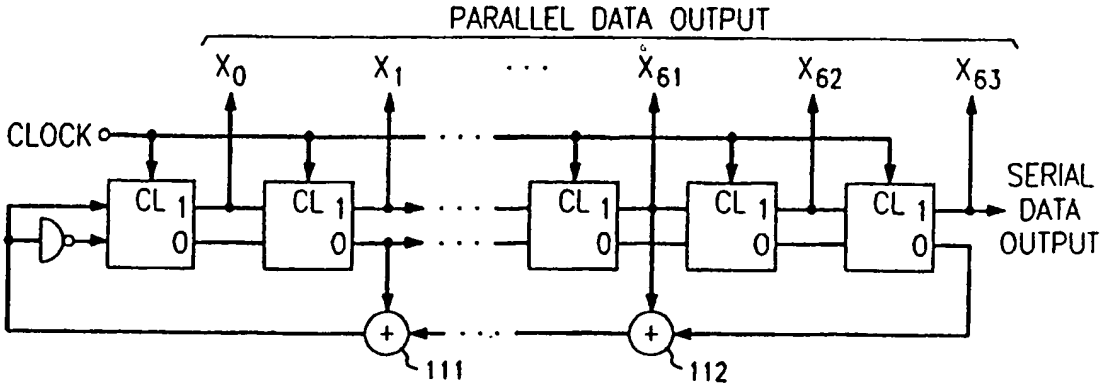


FIG. 11

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 427 465 A3**

(12) EUROPEAN PATENT APPLICATION

(21) Application number: **90312005.3**

(22) Date of filing: **01.11.90**

(51) Int. Cl.5: **G07F 7/10**, G07C 9/00, G06F 1/00

(30) Priority: **09.11.89 US 433821**

(43) Date of publication of application:
15.05.91 Bulletin 91/20

(84) Designated Contracting States:
DE FR GB IT

(88) Date of deferred publication of the search report:
24.07.91 Bulletin 91/30

(71) Applicant: **AMERICAN TELEPHONE AND TELEGRAPH COMPANY**
550 Madison Avenue
New York, NY 10022(US)

(72) Inventor: **Claus, David Michael**
7660 Brookview Lane
Indianapolis, Indiana 46250(US)

Inventor: **Coutinho, Roy S.**
10905 Timber Lane
Carmel, Indiana 46032(US)
Inventor: **Murphy, Kevin Dean**
6021 Middle Drive
Indianapolis, Indiana 46236(US)
Inventor: **Snavley, James Damon**
262 North Brewer Street
Greenwood, Indiana 46142(US)
Inventor: **Zempol, Kenneth Robert**
44 Center Grove Road, Apt. F26
Randolph, New Jersey 07920(US)

(74) Representative: **Watts, Christopher Malcolm Kelway et al**
AT&T (UK) LTD. AT&T Intellectual Property Division 5 Mornington Road
Woodford Green Essex IG8 OTU(GB)

(54) Databaseless security system.

(57) An improved security system, including a portable smart card (500) and a host computer (600), eliminates the need for the computer to store individual personal identification (ID) numbers for each user seeking access to the computer. Instead, the computer stores a first encryption algorithm E_1 used in converting a particular identification number $(ID)_n$ into a secret code S_n for that particular user. S_n also exists within the memory of the smart card having been loaded into its memory at the time of issue. A challenge number C is generated by the computer and transmitted to the smart card. Within the smart card and the computer, microprocessors respond to the challenge number C, the secret code S_n, and a second encryption algorithm E_2 in order to generate response numbers R_n and R'_n respectively. Thereafter, R_n is transmitted to the computer where it is compared with R'_n. A favorable comparison is necessary for gaining access to the computer.

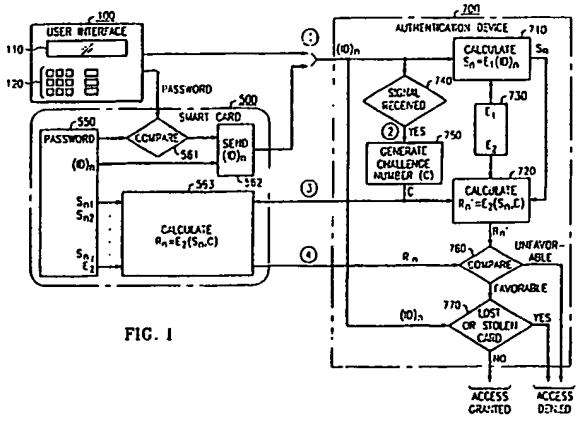


FIG. 1

EP 0 427 465 A3

European Patent Office

EUROPEAN SEARCH REPORT

Application Number

EP 90 31 2005

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int. Cl.5)
A	EP-A-0 029 894 (IBM) * abstract; claim 1 *	1,9	G 07 F 7/10 G 07 C 9/00
A	EP-A-0 131 421 (AMERICAN TELEPHONE AND TELEGRAPH COMPANY) * abstract *	1,9	G 06 F 1/00
A	EP-A-0 281 059 (SIEMENS) * abstract *	1,9	
A	US-A-4 310 720 (CHECK) * abstract *	1,9	
A	EP-A-0 114 773 (CII HONEYWELL BULL) * abstract *	1,9	
A	EP-A-0 281 058 (SIEMENS) * abstract *	1,9	
A	EP-A-0 284 133 (TRT) * claim 1 *	1,9	
D,A	US-A-4 453 074 (WEINSTEIN)		
D,A	US-A-4 471 216 (HERVE) * abstract *	1,9	

TECHNICAL FIELDS SEARCHED (Int. Cl.5)
G 07 F G 07 C G 06 F

The present search report has been drawn up for all claims

Place of search	Date of completion of search	Examiner
The Hague	30 May 91	TACCOEN J-F.P.L.

CATEGORY OF CITED DOCUMENTS

- X: particularly relevant if taken alone
- Y: particularly relevant if combined with another document of the same catagory
- A: technological background
- O: non-written disclosure
- P: intermediate document
- T: theory or principle underlying the invention
- E: earlier patent document, but published on, or after the filing date
- D: document cited in the application
- L: document cited for other reasons

& : member of the same patent family, corresponding document

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 427 465 A3**

(12) EUROPEAN PATENT APPLICATION

(21) Application number: 90312005.3

(22) Date of filing: 01.11.90

(51) Int. Cl.5: G07F 7/10, G07C 9/00, G06F 1/00

(30) Priority: 09.11.89 US 433821

(43) Date of publication of application: 15.05.91 Bulletin 91/20

(84) Designated Contracting States: DE FR GB IT

(88) Date of deferred publication of the search report: 24.07.91 Bulletin 91/30

(71) Applicant: AMERICAN TELEPHONE AND TELEGRAPH COMPANY
550 Madison Avenue
New York, NY 10022(US)

(72) Inventor: Claus, David Michael
7660 Brookview Lane
Indianapolis, Indiana 46250(US)

Inventor: Coutinho, Roy S.
10905 Timber Lane
Carmel, Indiana 46032(US)
Inventor: Murphy, Kevin Dean
6021 Middle Drive
Indianapolis, Indiana 46236(US)
Inventor: Snavley, James Damon
262 North Brewer Street
Greenwood, Indiana 46142(US)
Inventor: Zempol, Kenneth Robert
44 Center Grove Road, Apt. F26
Randolph, New Jersey 07920(US)

(74) Representative: Watts, Christopher Malcolm Kelway et al
AT&T (UK) LTD. AT&T Intellectual Property Division 5 Mornington Road
Woodford Green Essex IG8 OTU(GB)

(54) Databaseless security system.

(57) An improved security system, including a portable smart card (500) and a host computer (600), eliminates the need for the computer to store individual personal identification (ID) numbers for each user seeking access to the computer. Instead, the computer stores a first encryption algorithm E_1 used in converting a particular identification number $(ID)_n$ into a secret code S_n for that particular user. S_n also exists within the memory of the smart card having been loaded into its memory at the time of issue. A challenge number C is generated by the computer and transmitted to the smart card. Within the smart card and the computer, microprocessors respond to th challenge number C, the s cret cod S_n, and a second ncryption algorithm E_2 in order to generat r spons numbers R_n and R'_n respectiv ly. Th reafter, R_n is transmitted to the computer where it is compared with R'_n. A favorable comparison is necessary for gaining access to the computer.

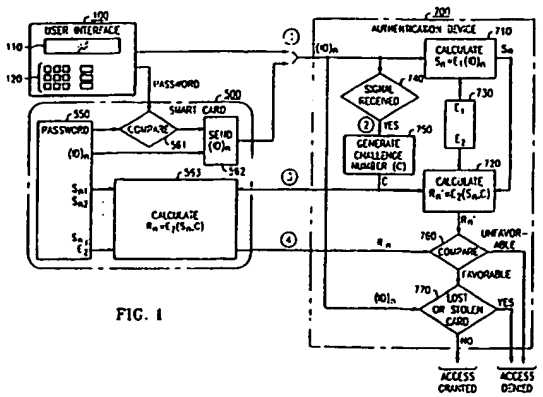


FIG. 1

EP 0 427 465 A3

European Patent Office

EUROPEAN SEARCH REPORT

Application Number

EP 90 31 2005

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int. Cl.5)
A	EP-A-0 029 894 (IBM) * abstract; claim 1 *	1,9	G 07 F 7/10 G 07 C 9/00 G 06 F 1/00
A	EP-A-0 131 421 (AMERICAN TELEPHONE AND TELEGRAPH COMPANY) * abstract *	1,9	
A	EP-A-0 281 059 (SIEMENS) * abstract *	1,9	
A	US-A-4 310 720 (CHECK) * abstract *	1,9	
A	EP-A-0 114 773 (CII HONEYWELL BULL) * abstract *	1,9	
A	EP-A-0 281 058 (SIEMENS) * abstract *	1,9	
A	EP-A-0 284 133 (TRT) * claim 1 *	1,9	
D,A	US-A-4 453 074 (WEINSTEIN)		TECHNICAL FIELDS SEARCHED (Int. Cl.5) G 07 F G 07 C G 06 F
D,A	US-A-4 471 216 (HERVE) * abstract *	1,9	

The present search report has been drawn up for all claims

Place of search	Date of completion of search	Examiner
The Hague	30 May 91	TACCOEN J-F.P.L.

CATEGORY OF CITED DOCUMENTS

- X: particularly relevant if taken alone
- Y: particularly relevant if combined with another document of the same catagory
- A: technological background
- O: non-written disclosure
- P: intermediate document
- T: theory or principle underlying the invention
- E: earlier patent document, but published on, or after the filing date
- D: document cited in the application
- L: document cited for other reasons

&: member of the same patent family, corresponding document

(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) N° de publication : **2 667 419**
(à n'utiliser que pour les commandes de reproduction)

(21) N° d'enregistrement national : **90 12114**

(51) Int Cl⁵ : G 06 K 19/073

(12) DEMANDE DE BREVET D'INVENTION A1

(22) **Date de dépôt** : 02.10.90.

(30) **Priorité** :

(43) **Date de la mise à disposition du public de la demande** : 03.04.92 Bulletin 92/14.

(56) **Liste des documents cités dans le rapport de recherche** : *Se reporter à la fin du présent fascicule.*

(60) **Références à d'autres documents nationaux apparentés** :

(71) **Demandeur(s)** : *Société dite GEMPLUS CARD INTERNATIONAL Société Anonyme* — FR.

(72) **Inventeur(s)** : Sourenian Paul et Geronimi François.

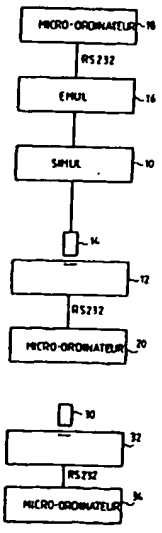
(73) **Titulaire(s)** :

(74) **Mandataire** : Cabinet Ballot-Schmit.

(54) Procédé de débogage de programme d'application de carte à mémoire et système de débogage.

(57) L'invention concerne les systèmes de débogage de programmes d'application de cartes à puces. Elle s'applique aux cartes comportant un microprocesseur et une mémoire non volatile programmable électriquement, cette mémoire contenant un programme d'application exécutable par le microprocesseur.

Pour réaliser le débogage on utilise un système très simple fondé sur le fait que la mémoire du programme d'application est programmable électriquement. on se sert de cette mémoire pour stocker d'une part des versions provisoires et modifiables du programme d'application, et d'autre part un programme d'aide au débogage. Le système de débogage comprend en pratique seulement une carte à puce échantillon (30), un lecteur de carte (32), et un micro-ordinateur (34) pour piloter les échanges entre la carte et le lecteur. Le procédé consiste essentiellement à stocker en mémoire le programme à déboguer, à modifier une instruction de ce programme pour la remplacer par une instruction de branchement vers le programme d'aide au débogage, et à lancer le programme d'application. Le programme d'aide fournit des renseignements sur l'état du système au moment du branchement.



FR 2 667 419 - A1

PROCEDE DE DEBOGAGE DE PROGRAMME D'APPLICATION DE CARTE A MEMOIRE ET SYSTEME DE DEBOGAGE

L'invention concerne la mise au point de programmes d'application de cartes à puces incorporant un microprocesseur et une mémoire de programmes.

Si la carte à puce comprend un microprocesseur, c'est pour pouvoir exécuter des programmes d'application qui se présentent sous forme d'instructions successives données au microprocesseur. Ces instructions portent sur des opérations internes à la carte et sur les échanges de données effectués entre la carte et un lecteur de carte dans lequel la carte est insérée.

Pour la mise au point de tout programme d'application de microprocesseur, il est nécessaire de passer par une phase de débogage dans laquelle le programme d'application est testé, dans des conditions aussi proches que possible de la réalité, et les erreurs éventuelles sont détectées puis corrigées. Les cartes à puces à microprocesseur n'échappent pas à cette règle.

Les cartes à puces qui sont plus spécialement concernées par la présente invention sont les cartes à microprocesseur comportant non seulement une mémoire morte (ROM) et une mémoire vive de travail (RAM), mais aussi une mémoire électriquement effaçable et reprogrammable (EEPROM) ou simplement électriquement programmable (EPROM). La mémoire morte comprend un programme figé, intangible, représentant notamment le système d'exploitation de la carte à microprocesseur : gestion des mémoires, gestion des sécurités d'accès, et plus généralement tous les programmes obligatoires non modifiables de la carte à puce. La mémoire RAM sert classiquement à stocker des données temporairement et de

manière volatile au cours de l'exécution d'un programme. La mémoire EEPROM ou EPROM contient des données non volatiles et peut contenir aussi des programmes d'application variés, pour que la carte puisse avoir des fonctionnalités supplémentaires, spécifiques d'une application donnée.

L'invention s'intéresse tout particulièrement au débogage des programmes d'application stockés en mémoire non volatile EEPROM ou EPROM et exécutables directement à partir de cette mémoire.

Dans la suite on ne parlera que de mémoires EEPROM, c'est-à-dire non seulement programmables mais aussi effaçables électriquement, l'invention étant particulièrement intéressante dans ce cas.

Pour effectuer le débogage d'un programme d'application qui est ainsi destiné à être stocké en mémoire non volatile pour être exécuté, on utilise habituellement un outil de développement qui comprend d'une part un simulateur de carte et d'autre part un émulateur de microprocesseur.

La figure 1 rappelle la constitution classique d'un outil de développement d'application de microprocesseur.

Le simulateur de carte 10 est un appareil électronique destiné à être connecté d'une part à un lecteur de carte 12 (par l'intermédiaire d'une extension 14 simulant les contacts d'accès d'une carte à puce), et d'autre part à l'émulateur 16. Le simulateur remplace la carte et possède toutes les ressources de la carte (mémoires RAM, ROM, EEPROM, interfaces d'entrée/sortie etc.) sauf le microprocesseur.

L'émulateur 16 simule le fonctionnement du microprocesseur de la carte et comprend à cet effet un microprocesseur identique à celui des cartes utilisées dans l'application. L'émulateur est relié d'une part au

simulateur de carte 10 pour que son microprocesseur soit dans le même environnement que s'il était monté effectivement sur une carte avec les ressources de celle-ci; et il est relié d'autre part à un microordinateur 18 capable de le commander c'est-à-dire de lui fournir toute instruction souhaitée et capable d'échanger des données avec lui.

Le lecteur de carte 12 quant à lui est relié aussi à un autre microordinateur 20 capable de le commander pour lui permettre d'échanger des données avec le simulateur de carte par l'intermédiaire de l'extension 14, exactement comme si une véritable carte était insérée dans le lecteur. C'est par ce deuxième microordinateur 20 que l'on pourra recueillir toutes les informations utiles sur ce qui se passe dans la carte, le premier microordinateur ayant pour rôle de fournir des instructions au microprocesseur pour lui faire exécuter des programmes contrôlés.

Le développement de l'application avec cet outil classique consiste à :

- faire exécuter un programme d'application par le microprocesseur de l'émulateur, soit complètement soit partiellement, soit en mode pas-à-pas, soit en imposant des points d'arrêts, etc.

- examiner le fonctionnement global et détaillé, en s'intéressant notamment aux contenus des mémoires et registres internes du microprocesseur à diverses étapes du programme.

- détecter les défauts de fonctionnalités et autres erreurs;

- modifier le programme d'application dans un sens tendant à supprimer les défauts constatés;

- et recommencer le débogage avec le programme d'application modifié, ceci jusqu'à édition complète

des défauts constatés.

Il faut donc un outil de développement relativement compliqué (un émulateur + un simulateur + deux ordinateurs personnels) et donc cher pour effectuer ce débogage.

D'autre part le simulateur de carte est rarement parfaitement compatible avec la carte qu'il doit simuler : il peut y avoir des écarts de conditions d'opération (tension d'alimentation, fréquence d'horloge, jeu d'instructions, etc.). En effet, on ne modifie pas les outils de débogage, émulateur et simulateur, chaque fois qu'on apporte des petites modifications aux séries de cartes fabriquées.

Enfin, l'utilisateur de la carte, qui développe l'application, n'est pas le fabricant de la carte et du matériel de simulation et il peut être à l'autre bout de la planète. La maintenance de l'outil de développement peut donc poser des problèmes difficiles.

Un but de l'invention est d'améliorer les outils de débogage pour les cartes à puces à microprocesseur incorporant une mémoire non volatile programmable électriquement, susceptible de contenir un programme d'application.

Selon l'invention, on propose un système de débogage de programme d'application de carte à mémoire comprenant une carte à mémoire échantillon correspondant à celle utilisée dans l'application à déboguer, cette carte comportant au moins un microprocesseur et une mémoire non volatile programmable électriquement, un lecteur de carte dans lequel cette carte est introduite, un microordinateur (ordinateur personnel ou autre console de contrôle) pour contrôler le lecteur et notamment les échanges de données entre la carte et le lecteur, le programme à déboguer étant contenu dans une

première zone de la mémoire non volatile de la carte, et un programme d'aide au débogage étant contenu dans une deuxième zone de la mémoire non volatile.

En pratique, le programme d'aide au débogage comporte des moyens pour sauvegarder dans la mémoire non volatile des données internes représentant le fonctionnement de la carte à un instant donné.

On peut notamment charger dans la mémoire non volatile un programme d'application comportant une instruction modifiée à l'endroit où on veut observer le comportement interne de la carte, cette instruction modifiée étant une instruction de branchement vers le programme d'aide au débogage.

Le procédé de débogage selon l'invention comprend les opérations consistant à :

- introduire une carte échantillon dans un lecteur de cartes contrôlé par un microordinateur, la carte comportant un microprocesseur et une mémoire non volatile programmable électriquement,
- charger dans la mémoire non volatile, à l'aide du microordinateur et du lecteur, un programme d'application à déboguer, et un programme d'aide au débogage,
- donner par l'intermédiaire du microordinateur et du lecteur des ordres de modification et/ou d'exécution du programme chargé, l'exécution étant éventuellement modifiée sous le contrôle du programme d'aide au débogage, et recueillir dans le microordinateur les données résultant de l'exécution du programme,
- modifier le programme d'application s'il y a lieu en fonction des résultats et recommencer les étapes de chargement d'un programme d'application, d'exécution, et de recueil des données.

Le programme d'aide au débogage pourra comporter des moyens pour simuler la suite d'une séquence d'échanges entre la carte et le microprocesseur lorsqu'une telle séquence a été interrompue en vue de la sauvegarde du contexte machine dans la mémoire non volatile.

Par conséquent, au lieu d'utiliser un simulateur de carte qui comprend toutes les ressources de la carte sauf le microprocesseur, un émulateur qui comprend le microprocesseur, et un microordinateur qui comprend un programme pour le microprocesseur, on utilise simplement une carte échantillon, identique à celle qui sera effectivement utilisée dans l'application. Cette carte comprend ses propres ressources (mémoires vives, mortes et non volatiles), son microprocesseur, un programme d'application stocké en mémoire non volatile et un programme d'aide au débogage stocké dans la même mémoire et apte à modifier partiellement l'exécution du programme d'application. Seul un lecteur de carte et un seul microordinateur pour le contrôler sont nécessaires pour effectuer le débogage. Le débogage s'effectue par chargements successifs de programmes modifiés jusqu'à élimination complète des erreurs. On utilise ainsi très avantageusement le fait que le programme d'application est directement exécutable à partir d'une mémoire non volatile et le fait que cette mémoire est programmable électriquement à partir du lecteur de cartes.

Le microordinateur est capable notamment d'insérer des points d'arrêt (instructions de branchement vers le programme d'aide au débogage) à des points désignés du programme d'application. Le programme d'aide au débogage stocké en mémoire non volatile est de préférence capable de lire et modifier les contenus des diverses zones de mémoire (vive et non volatile) et de sauvegarder en

mémoire non volatile les contenus de registres divers de la carte, cela à n'importe quel moment du processus d'échange entre la carte et le lecteur.

D'autres caractéristiques et avantages de l'invention apparaîtront à la lecture de la description détaillée qui suit et qui est faite en référence aux dessins dans lesquels :

- la figure 1, déjà décrite, représente la constitution classique d'un outil de développement d'application de carte à puce à microprocesseur;
- la figure 2 représente la constitution de l'outil de développement selon l'invention;
- la figure 3 représente schématiquement le contenu de la puce d'une carte à puce;
- la figure 4 représente l'organisation de la mémoire non volatile avec des zones de données et des zones de codes exécutables;
- la figure 5 représente un exemple de séquence d'échanges entre une carte à puce et un lecteur de carte.

La figure 2 représente l'outil de développement selon l'invention permettant le débogage d'un programme d'application d'une carte à microprocesseur lorsque ce programme est stocké dans une mémoire non volatile interne, programmable électriquement (en général EEPROM donc également effaçable électriquement), de la puce de circuit intégré de la carte.

L'outil est très simple : il comprend

- une carte échantillon 30 correspondant exactement aux cartes qui seront utilisées lorsque le programme sera définitivement au point;
- un lecteur de carte 32 dans lequel on peut insérer la carte et échanger avec elle des données selon une procédure qui n'est pas différente de la procédure qu'on utilisera dans l'application finale;

- et un microordinateur 34 pour contrôler le lecteur et donc les échanges avec la carte. Les échanges avec la carte sont classiquement des échanges en série. Les échanges entre le microordinateur et le lecteur peuvent se faire par une liaison classique type RS232.

La carte échantillon, comme la carte de l'application finale comporte classiquement, comme cela est représenté à la figure 3 un microprocesseur (CPU) avec tous ses registres et opérateurs internes, une mémoire morte ROM de programme, pour les programmes figés déjà mis au point et principalement pour le système d'exploitation de la carte, une mémoire vive de travail (RAM), volatile, et une mémoire non volatile programmable électriquement, de préférence EEPROM, pouvant contenir des données mais aussi du code exécutable, notamment le programme d'application qu'on cherche à mettre au point. La carte peut comprendre bien sûr d'autres circuits et notamment des circuits de sécurité établissant les autorisations d'échanges entre le lecteur et la carte. Ces échanges se font par des connexions d'entrée/sortie E/S, en principe en mode série.

Lors de la mise au point du programme d'application contenu en mémoire non volatile, on utilise une première zone Z1 de cette mémoire pour contenir le programme à déboguer, et une autre zone de mémoire Z2 pour contenir un programme d'aide au débogage. Ces deux zones contiennent du code exécutable par le microprocesseur de la carte. Une troisième zone Z3 sert de mémoire de données non volatiles; ces trois zones sont accessibles en lecture, écriture et effacement par le microprocesseur. D'autres zones peuvent encore être prévues, par exemple pour contenir des informations non accessibles en lecture et/ou en programmation et/ou en effacement par le microprocesseur.

La figure 4 représente cette organisation de la mémoire non volatile de la carte échantillon qui sert au débogage.

Le programme d'aide au débogage qui est contenu dans la zone Z2 de la mémoire est un programme permettant de faire exécuter par le microprocesseur diverses opérations permettant d'analyser le déroulement du programme principal; par exemple, ce programme d'aide comprend des instructions permettant de stocker en mémoire non volatile les contenus de registres et de la mémoire vive du microprocesseur, puis de les lire pour les envoyer au microordinateur. Il permet aussi de lire et modifier des zones de la mémoire vive ou de la mémoire non volatile.

La mise au point du programme se déroule de la manière suivante : par le microordinateur 34 et par l'intermédiaire du lecteur de carte 32 commandé par le microordinateur, on charge dans la zone Z1 de la mémoire non volatile de la carte le programme d'application; on modifie par exemple une instruction de ce programme pour y introduire un point d'arrêt (permettant d'examiner l'état de la carte à l'instant où ce point est atteint); le point d'arrêt consiste en une instruction de branchement vers le programme d'aide au débogage qu'on a stocké dans la zone Z2.

On fait exécuter le programme d'application jusqu'à l'instruction de branchement, à partir de laquelle le programme d'aide au débogage prend le relais pour effectuer certaines opérations, et notamment le relevé de certains registres et mémoires du circuit. Ces relevés sont stockés en mémoire EEPROM, dans la zone de données Z3. Ces relevés seront lus ultérieurement par le microordinateur 34; ils permettent de contrôler le fonctionnement de la carte et de détecter des erreurs de fonctionnement.

Le programme d'aide au débogage permet aussi de modifier le contenu de certaines zones de mémoire vive ou de la mémoire non volatile, et même le contenu de certains registres internes du microprocesseur.

Dans la pratique, les normes d'utilisation des cartes à puces prévoient un protocole d'échanges assez strict entre la carte et le lecteur.

Ce protocole défini par la norme ISO 7816-3 est encore appelé T = 0 ; il est représenté schématiquement à la figure 5 : c'est le lecteur de carte qui a le contrôle permanent des échanges et non la carte (pendant la mise au point de l'application, le lecteur est contrôlé par le microordinateur 34); le lecteur envoie des octets de commande CMD pour définir les opérations à effectuer par la carte; la carte répond par un octet de procédure PB; puis les échanges de données ont lieu (signaux DATA IN ou DATA OUT selon que la carte reçoit ou émet des données); et enfin la carte émet deux octets de fin de procédure ME1, ME2. Si le déroulement de la séquence n'est pas conforme à cette procédure, le lecteur envoie un message d'erreur et reprend le contrôle. Cette procédure normalisée est donnée à titre d'exemple.

Une des fonctions importantes du débogage est la connaissance du contexte machine (contenu de la mémoire vive, contenu des registres microprocesseur) à un instant donné du déroulement du programme d'application; cela veut dire qu'une séquence d'échanges entre le lecteur et la carte sera forcément interrompue si on a besoin de connaître le contexte au milieu de cette séquence. Or le lecteur réagit à toute rupture de séquence en émettant un message d'erreur et en réinitialisant tous les registres.

Selon l'invention, on propose un moyen pour permettre quand même la connaissance du contexte machine

à un instant désiré d'une séquence d'échanges.

Pour atteindre ce but, le programme d'aide au débogage (vers lequel on déroute le programme d'application à l'endroit désiré) contient des instructions de sauvegarde des contenus registres et mémoires qu'on veut observer, et des instructions de simulation d'échanges de données avec le lecteur. La sauvegarde est faite dans la zone de données Z3 de la mémoire EEPROM.

Par conséquent, la séquence d'analyse se déroule de la manière suivante : le programme d'application est chargé dans la mémoire EEPROM (zone Z1) avec une instruction modifiée à l'endroit où on veut observer le contexte machine; l'instruction modifiée est une instruction de branchement vers le programme d'aide au débogage (zone Z2); le programme d'aide au débogage écrit alors en mémoire EEPROM (zone Z3) le contenu des registres et mémoires et il exécute une séquence fictive d'échanges avec le lecteur de carte (à partir du point où cette séquence a été interrompue) pour laisser croire au lecteur que la séquence en cours se déroule normalement.

Par exemple, si la séquence en cours est une séquence d'envoi de données par la carte et si on veut observer le contexte machine juste après l'octet de procédure PB, le programme d'aide au débogage enverra des données quelconques au lecteur pendant la sauvegarde du contexte machine. Si la séquence en cours était une séquence de réception de données par la carte, le programme d'aide au débogage simulerait une réception de données.

Le contexte machine sauvegardé en mémoire non volatile à des adresses spécifiées de la zone Z3 pourra être lu ultérieurement : le microordinateur 34 peut lire ou écrire à n'importe quelle adresse de la mémoire non

volatile sauf éventuellement dans des zones à accès réservé. La révélation du contenu des registres du microprocesseur par la lecture ultérieure de la zone Z3 permet de comprendre les erreurs du programme de l'application et de les corriger.

Par exemple, pour le microprocesseur ST8 de SGS-THOMSON MICROELECTRONICS S.A., la série d'instructions microprocesseurs de débogage qui servent à la sauvegarde du contenu des registres, est, en langage assembleur, la suivante :

```
                Input  Filename : BPHGR.asm
                Output Filename : BPHGR.obj

55                              ****************************
56                              *  TSA: TRANSFERT SP TO A  *
57                              ****************************
58                              TSA  MACRO
59                                   FCB   $9E
60                                   ENDM
61
62                              ********************************************************************************
63   E600                             ORG $E600
64   E600                       BACKUP    BLKB 40,0          * ZONE DE SAUVEGARDE DE LA RAM ET DES
65                                                           * REGISTRES A, X, CC ET SP
66
67   E628  24 04                CSET      BHS   CCLR          *
68   E62A  10 0A                          BSET  0,REG_CC     *
69   E62C  20 02                          BRA   ZSET         * RECUPERATION DE CC,A ET X
70   E62E  11 8A                CCLR      BCLR  0,REG_CC     *
71                                                           *
72   E630  26 04                ZSET      BNE   ZCLR         *
73   E632  12 8A                          BSET  1,REG_CC     *
74   E634  20 02                          BRA   NSET         *
75   E636  13 0A                ZCLR      BCLR  1,REG_CC     *
76                                                           *
77   E638  2A 04                NSET      BPL   NCLR         *
78   E63A  14 8A                          BSET  2,REG_CC     *
79   E63C  20 02                          BRA   ISET         *
80   E63E  15 0A                NCLR      BCLR  2,REG_CC     *
81                                                           *
82   E640  2C 04                ISET      BHC   ICLR         *
83   E642  16 8A                          BSET  3,REG_CC     *
84   E644  20 02                          BRA   HSET         *
85   E646  17 8A                ICLR      BCLR  3,REG_CC     *
86                                                           *
87   E648  28 04                HSET      BHCC  HCLR         *
88   E64A  18 8A                          BSET  4,REG_CC     *
89   E64C  20 02                          BRA   NXT_REG      *
90   E64E  19 8A                HCLR      BCLR  4,REG_CC     *
91                                                           *
92   E650  B7 88                NXT_REG   STA   REG_A        *
93   E652  BF 89                          STX   REG_X        *
94   E654                                 TSA                *
95   E654  9E                        FCB   $9E
96   E655                            ENDM

97   E655  B7 8B                          STA   REG_SP       *
```

Cette série d'instructions est utilisable telle quelle sur tous les microprocesseurs du type 6805. Elle doit être modifiée en fonction du langage utilisé par le microprocesseur, pour les autres microprocesseurs. Sa structure et son but se déduisent de ceux indiqués ci-dessus.

REVENDICATIONS

1. Système de débogage de programme d'application de carte à mémoire comprenant une carte à mémoire échantillon (30) correspondant à celle utilisée dans l'application à déboguer, cette carte comportant au moins un microprocesseur et une mémoire non volatile programmable électriquement, un lecteur de carte (32) dans lequel cette carte est introduite, un microordinateur (34) pour contrôler le lecteur et notamment les échanges de données entre la carte et le lecteur, le programme à déboguer étant contenu dans une première zone (Z1) de la mémoire non volatile de la carte, et un programme d'aide au débogage étant contenu dans une deuxième zone (Z2) de la mémoire non volatile.

2. Système selon la revendication 1, caractérisé en ce que le programme d'aide au débogage comporte des moyens pour sauvegarder dans la mémoire non volatile des données internes représentant le fonctionnement de la carte.

3. Système selon l'une des revendications 1 et 2, caractérisé en ce qu'il comporte des moyens pour charger dans la mémoire non volatile un programme d'application comportant une instruction modifiée à l'endroit où on veut observer le comportement interne de la carte, cette instruction modifiée étant une instruction de branchement vers le programme d'aide au débogage.

4. Procédé de débogage d'application de carte à puce, ce procédé étant caractérisé en ce qu'il comprend les opérations consistant à :

- introduire une carte échantillon (30) dans un lecteur de cartes (32) contrôlé par un microordinateur (34), la carte comportant un

microprocesseur et une mémoire non volatile programmable électriquement,

- charger dans la mémoire non volatile, à l'aide du microordinateur et du lecteur, un programme d'application à déboguer, et un programme d'aide au débogage,

- donner par l'intermédiaire du microordinateur et du lecteur des ordres de modification et/ou d'exécution du programme chargé, l'exécution étant éventuellement modifiée sous le contrôle du programme d'aide au débogage, et recueillir dans le microordinateur les données résultant de l'exécution du programme,

- modifier le programme d'application s'il y a lieu en fonction des résultats et recommencer les étapes de chargement d'un programme d'application, d'exécution, et de recueil des données.

5. Procédé selon la revendication 4, caractérisé en ce que le programme d'aide au débogage comporte des moyens pour sauvegarder dans la mémoire non volatile les contenus de mémoires et registres internes représentant le contexte machine du microprocesseur.

6. Procédé selon la revendication 5, caractérisé en ce que le programme d'aide au débogage comporte des moyens pour simuler la suite d'une séquence d'échanges entre la carte et le microprocesseur lorsqu'une telle séquence a été interrompue en vue de la sauvegarde du contexte machine dans la mémoire non volatile.

FIG_1

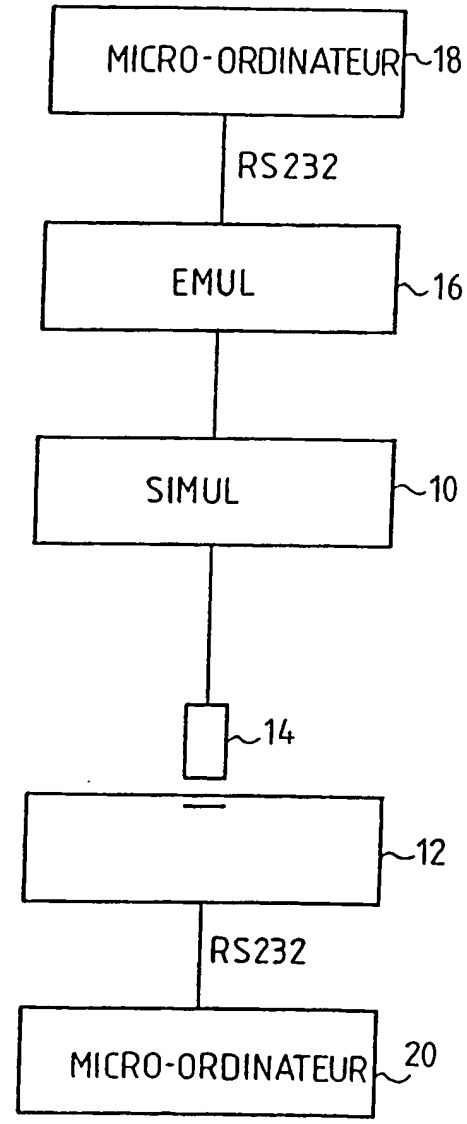

MICRO-ORDINATEUR
18
RS232
EMUL
16
SIMUL
10
14
12
RS232
MICRO-ORDINATEUR
20

FIG_2

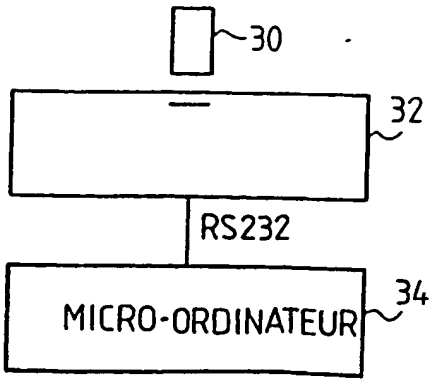
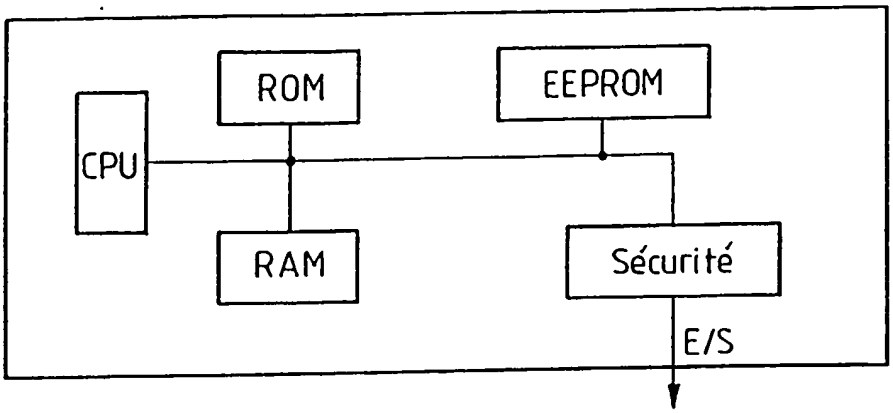

30
32
RS232
MICRO-ORDINATEUR
34

FIG_3



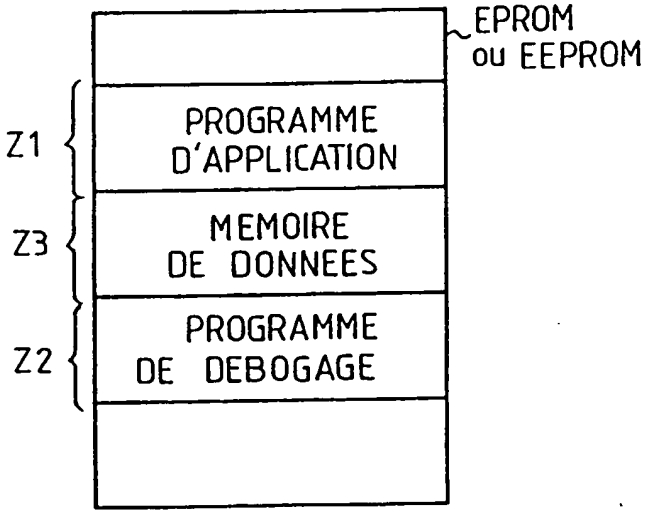
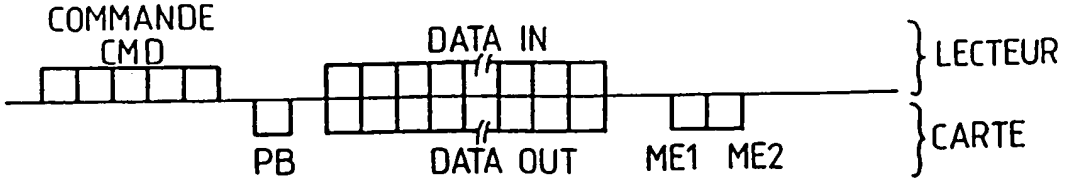


FIG_4

FIG_5



REPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
de la
PROPRIETE INDUSTRIELLE

2667419
N° d'enregistrement national

RAPPORT DE RECHERCHE

établi sur la base des dernières revendications déposées avant le commencement de la recherche

FR 9012114
FA 448145

DOCUMENTS CONSIDERES COMME PERTINENTS

Catégorie	Citation du document avec indication, en cas de besoin, des parties pertinentes	Revendications concernées de la demande examinée
X	EP-A-0 356 237 (HITACHI) * Abrégé; colonne 2, lignes 30-65; colonne 3, lignes 1-15: colonne 3, lignes 1-57 *	1-6
A	US-A-4 777 355 (MITSUBISHI) * Abrégé *	1,2
A	FR-A-2 633 755 (MITSUBISHI) * Abrégé *	1,3

DOMAINES TECHNIQUES RECHERCHES (Int. Cl.5)

G 06 K
G 07 F

Date d'achèvement de la recherche: 03-05-1991

Examinateur: CHIARIZIA S.J

CATEGORIE DES DOCUMENTS CITES

X : particulièrement pertinent à lui seul
Y : particulièrement pertinent en combinaison avec un autre document de la même catégorie
A : pertinent à l'encontre d'au moins une revendication ou arrière-plan technologique général
O : divulgation non-écrite
P : document intercalaire

T : théorie ou principe à la base de l'invention
E : document de brevet bénéficiant d'une date antérieure à la date de dépôt et qui n'a été publié qu'à cette date de dépôt ou qu'à une date postérieure.
D : cité dans la demande
L : cité pour d'autres raisons

& : membre de la même famille, document correspondant

EPO FORM 1503 03.82 (P0413)